Aprendizaje Jerárquico por Refuerzo mediante Programación Genética dirigida por Gramáticas

S. García, L. Sánchez

Resumen— **En este trabajo describimos una metodología, basada en el empleo de la Programación Genética Difusa dirigida por gramáticas, aplicable a problemas de Aprendizaje por Refuerzo con naturaleza jerárquica. Con esta técnica puede obtenerse una estrategia de control de un proceso industrial expresable lingüísticamente. Se supondrá que se pueden utilizar, de forma simultánea, dos tipos de información acerca del proceso: un modelo matemático matemático del mismo y conocimiento experto acerca del control. El conocimiento experto, que puede venir tanto en forma de reglas de comportamiento como de restricciones acerca de la estructura del propio sistema de control, se integrará en el algoritmo de aprendizaje como parte de la gramática con que se definen las soluciones válidas.**

En este trabajo se incluyen algunos resultados preliminares de la aplicación de esta metodología al diseño de un sistema de control distribuido de un problema industrial, y se detallan las partes del algoritmo sobre un problema de ejemplo.

Palabras clave— **Aprendizaje Jerárquico por Refuerzo, Programación Genética Difusa, Gramáticas.**

I. Introducción. Aprendizaje Jerárquico por Refuerzo.

EL Aprendizaje Jerárquico por Refuerzo se puede enfocar de distintas formas, dependiendo básicamente de las características del problema y del nivel de conocimiento existente sobre la solución óptima del mismo. Si bien existen distintas aproximaciones al tema de la extracción automática de conocimiento sobre problemas de naturaleza jerárquica [1], [2], [3], [4], todos los enfoques que conocemos se apoyan, en mayor o menor medida, en una descomposición del problema en subtareas, subobjetivos, o fases ordenadas secuencialmente que llevan a la resolución del problema. Esta descomposición hace necesaria la existencia de un grado de conocimiento acerca de la forma de resolver el problema de aprendizaje.

Una forma de incorporar conocimiento acerca de un problema de este tipo consiste en restringir la representación de las soluciones. En particular, puede emplearse una gramática para definir cuál es la estructura válida de una solución y aplicar un algoritmo de aprendizaje automático (como los algoritmos genéticos [5], el recocido simulado [6] u otros) para buscar la estrategia más adecuada dentro del conjunto definido por la gramática. La combinación de representación gramatical y búsqueda genética ha sido empleada anteriormente en inducción de descripciones lingüísticas; por ejemplo, en [7] se utilizó en el aprendizaje de sistemas de clasificación basados en reglas borrosas, cuya estructura tiene una cierta similitud con la de los problemas de aprendizaje por refuerzo que se tratarán en este trabajo.

Este estudio tiene como objeto el obtener descripciones lingüísticas de estrategias de control jerárquicas. Se partirá de un modelo del proceso que se desea controlar, y se dirigirá el aprendizaje hacia soluciones que cumplan restricciones tanto estructurales como de funcionamiento. En las siguientes secciones se expondrá la forma en la que, por una parte, las restricciones estructurales se incorporan en una gramática y, por otra, las restricciones de funcionamiento se integran en la función de adecuación de un individuo.

II. Objetivos del trabajo

En esta sección formalizaremos las características de los problemas a los que está orientada esta metodología. Partimos de una situación genérica en la que tenemos (ver figura 1):

- Un conjunto e_i de variables de estado continuas y otro de variables lógicas F_i.
- Un conjunto de actuadores, A_i
- Un conjunto de centros de control: C_i
- Las coordenadas de las variables de estado y de los centros de control: e_{ix}, e_{iy}, C_{ix}, y C_{iy}, respectivamente.
- Un conjunto de objetivos, expresados como predicados sobre las variables anteriores, y que desglosaremos en:
  - Objetivos que deben ser satisfechos de forma concurrente durante el funcionamiento del sistema, R_i. Serán interpretados como *restricciones* que deberá cumplir cualquier política de control viable.
  - Objetivos que deben ser satisfechos en la medida de lo posible, O_i.

Los resultados del problema de aprendizaje jerárquico definido de esta forma son (ver figura 2):

- Una Jerarquía de Control.
- El código de cada uno de los controladores.
- La caracterización de las funciones de pertenencia de todas las variables de estado.

III. Metodología.

Para inducir toda esta información a partir de la topología inicial, el algoritmo que proponemos consta de las siguientes fases:

Departamento de Informatica, Campus de Viesques, s/n, 33206 Gijon (España). e-mail: carbajal@lsi.uniovi.es

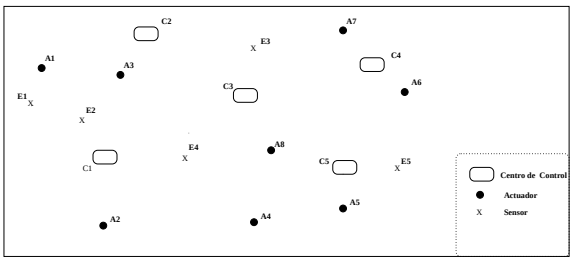

Fig. 1. Topología inicial de Variables de Control, Actuadores, y controladores Locales.

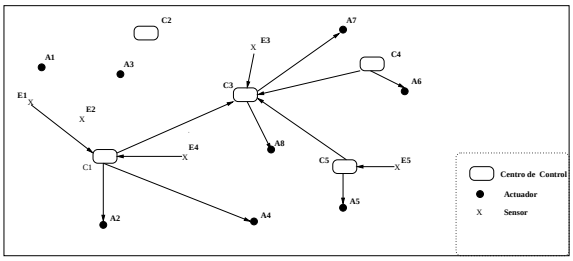

Fig. 2. Sistema de Control Jerárquico Inducido.

A. *Identificación de los elementos del sistema.*

A partir de las especificaciones del problema será necesario identificar y agrupar los elementos del sistema (sensores, actuadores, etc).

B. *Definición de la gramática.*

En el caso de contar únicamente con la información topológica referente a la ubicación física de los centros de control, sensores, y actuadores, y suponiendo la jerarquía de control desconocida, es posible aplicar el método denominado *Programación Genética Difusa* definido por Andreas Geyer Schultz [5], planteando una gramática que describa controladores difusos basados en bases de reglas. Para la inducción simultánea de las formas de las funciones de pertenencia de las etiquetas lingüísticas a las que hacen referencia las reglas, proponemos una adaptación del método GA-P [8], consistente en incluir una parte de la gramática destinada a almacenar en cada individuo la codificación de las funciones de pertenencia [7]. La gramática de un controlador difuso de este tipo se muestra en la figura 3.

Todo el proceso estará guiado, obviamente, por una función de fitness en la que estará presente tanto la calidad del control como el coste de implementación de la jerarquía obtenida. El problema a la hora de intentar inducir un controlador jerárquico sin restricciones es la excesiva dimensión del espacio de búsqueda y el elevado tiempo de convergencia necesario para obtener una solución aceptable. Es por esto que la gramática que describe los controladores jerárquicos candidatos a solución del problema debe permitir la inclusión del conocimiento existente sobre el mismo.

B.1 Representación de la topología de control.

La distribución física de los elementos del sistema es un factor importante en los problemas que hemos

```
S := <Controlador_Difuso_Jerárquico>;
<Controlador_Difuso_Jerarquico> := <Lista_de_Controladores>
<Lista_de_Constantes> ;
<Lista_de_Controladores> := <Controlador> |<Controlador>
<Lista_de_Controladores> ;
<Controlador> := <Lista_de_Reglas> ;
<Lista_de_Reglas> := <Regla> | <Regla> <Lista_de_Reglas>;
<Regla> := SI <Antecedente> ENTONCES <Consecuente> ;
<Antecedente>:= ( <Aserto>) | ( <Operador> <Antecedente>
<Antecedente> );
<Operador> := OR | AND ;
<Aserto> := <Variable> ES <Etiqueta> |
<Variable_Sintetizada> ;
<Consecuente> := <Lista_de_Acciones> <Variable_Sintetizada>;
<Lista_de_Acciones> := <Acción> | <Acción>
<Lista_de_Acciones>;
<Lista_de_Constantes> := <Constante> | <Constante>
<Lista_de_Constantes> ;
<Acción> := <Actuador> ES <Etiqueta>;
<Variable_sintetizada> :=<Síntesis> ES <Etiqueta> | NADA;
<Actuador> := A1 |A2|A3|.. |An;
<Variable> := e1|e2|e3 .. |em;
<Síntesis>:= S1|S2|.. |Sk;
<Etiqueta> := PEQUEÑO |MEDIANO |GRANDE;
<Constante>:= 1|2|3|4|5|6|7|8|9;
```

Fig. 3. Gramática genérica de un controlador difuso. m es el número de variables de entrada, n es el número de actuadores, y k es el número máximo de centros de control.

descrito. Por esta razón, el algoritmo que proponemos trabajará sobre una estructura de datos en la que esté almacenada la ubicación de cada elemento del sistema.

C. *Generación de la población inicial.*

Una forma de generar N cadenas del lenguaje podría ser construirlas de forma aleatoria a partir de los símbolos terminales de la gramática. Obviamente, un mecanismo de este tipo llevaría, en la mayor parte de los casos, a la obtención de individuos sintácticamente incorrectos, inviables como soluciones al problema dado. En los trabajos de Andreas Geyer Schultz [5] se describe un procedimiento de generación de la población inicial que construye cadenas de todas las profundidades posibles distribuyendo el número de individuos uniformemente. Tras algunas comparaciones experimentales, esta inicialización no ha producido buenos resultados, por lo que se ha optado por un mecanismo más sencillo:

1. Seleccionar un tamaño de población, N, y una profundidad máxima, P.
2. Seleccionar el símbolo inicial de la gramática, S.
3. Escoger aleatoriamente una producción del símbolo inicial, y sustituir por su parte derecha.
4. Repetir el proceso descrito en los dos puntos anteriores para cada no terminal aparecido como resultado de la sustitución, hasta alcanzar la profundidad límite.
5. Si la cadena está formada sólo por símbolos terminales, incluirla en la población.
6. Volver al paso 2 hasta alcanzar el tamaño de población escogido, P.

Las razones por las que creemos que este método ha dado buenos resultados, haciendo innecesario el algoritmo de generación de individuos descrito en [5] son las siguientes:

1. La profundidad media de la población crece de forma espontánea durante el proceso evolutivo.
2. La definición gramatical de los individuos hace que la búsqueda se plantee en un espacio de menores dimensiones que en un procedimiento de Programación Genética clásico.
3. Por otra parte, el procedimiento de generación

uniforme de la población inicial resulta tremendamente costoso computacionalmente hablando, por lo que se reserva su aplicación para casos en los que el proceso de búsqueda resulte infructuoso.

D. Operadores genéticos.

En los métodos de *Programación Genética Dirigida por Gramáticas* aparece un problema adicional a la hora de implementar los operadores genéticos: los individuos generados mediante estos mecanismos deben ser cadenas válidas dentro del lenguaje descrito por la gramática. Independientemente de que pudieran existir soluciones aceptables al problema que no cumplieran las restricciones impuestas por la gramática, el proceso de búsqueda debe centrarse en las que sí lo hacen. En este trabajo se han empleado los operadores definidos en [5].

E. Definición de la función de fitness.

En primer lugar definiremos una función genérica de fitness adaptada a las características del tipo de problema que hemos definido. A continuación, dos medidas adicionales de la calidad de un controlador, la modularidad y el solapamiento, con el objeto de analizar hasta qué punto es posible dirigir el proceso de búsqueda mediante la introducción de nuevos factores en la función de fitness.

En este trabajo hemos descrito problemas en los que hay que tener en cuenta los siguientes factores:

Condiciones de Seguridad: estarán definidas como predicados sobre las variables de estado del sistema.

Error de Control: definido como la integral del error de posición del sistema.

Coste de Control, C : será una medida del coste que implica la aplicación de un determinado controlador sobre la planta.

$$C = \int ||control^2||$$

Coste de Implementación, C_I : estimación del coste de implementación del controlador.

Dado un controlador jerárquico J, definimos la *Función de Fitness Gradual, $F_g(J)$* como:

$$\mathrm{F_g(J)} = \begin{cases} \text{Sistema fuera de márgenes de seguridad:} \\ -(\mathrm{C_I + C + E_c + (T - t_d) * (D + A)}) \\ \\ \text{Sistema dentro de márgenes de seguridad:} \\ -(\mathrm{C_I + C + E_c}) \end{cases}$$

donde

C_I es el coste de implementación.
C es el coste de control
E_c es la integral del error de posición del sistema.
T es la longitud del intervalo de simulación.
t_d es el momento en el que el sistema abandona los márgenes de seguridad.
D es la máxima desviación posible con respecto al estado objetivo.
A es la máxima acción de control posible en un instante determinado.

E.1 Modularidad

En un problema de naturaleza jerárquica, definiremos el *Coeficiente de Modularidad* de un Controlador como el grado de independencia entre los distintos controladores que lo componen.

Sea V_{C_i} el conjunto de variables que utiliza un controlador parcial C_i integrado en la jerarquía, y sea $|V|$ la cardinalidad de un conjunto V de variables de estado del sistema. Podemos definir el *Coeficiente de Modularidad* de un controlador jerárquico J, integrado por los controladores parciales C_i, como:

$$C_M(J) = \frac{|\cup V_{C_i}| - max\{|V_{C_i} \cap V_{C_j}|\}}{|\cup V_{C_i}|} \quad (1)$$

que alcanzará su valor máximo (1) cuando los controladores parciales no compartan ninguna variable.

La función de fitness adaptada a esta nueva situación es la siguiente:

$$\mathrm{F_{gm}(J)} = \begin{cases} \text{Sistema fuera de márgenes de seguridad:} \\ -(((\mathrm{C_I + C + E_c}) * (1 - \mathrm{C_M})) + (\mathrm{T - t_d}) * (\mathrm{D + A})) \\ \\ \text{Sistema dentro de los márgenes de seguridad:} \\ -(\mathrm{C_I + C + E_c}) * (1 - \mathrm{C_M}) \end{cases}$$

E.2 Solapamiento.

Los distintos controladores se activan en función del grado de pertenencia del estado del sistema, o del subconjunto de variables de estado con el que trabajan, a una serie de regiones difusas que se inducen durante el proceso de aprendizaje.

Sea V_{C_i} el conjunto de variables con las que trabaja C_i, y μ_{C_i} el grado de pertenencia de V_{C_i} a la región borrosa que activa el controlador C_i

El *Grado de Solapamiento* de un controlador jerárquico J, $G_S(J)$, se puede definir como:

$$G_S(J) = \frac{\int_t min\{(\mu_{C_i})\}dt}{\int_t max\{(\mu_{C_i})\}dt} \quad (2)$$

que se puede aproximar por:

$$\frac{\sum min\{(\mu_{C_i})\}}{\sum max\{(\mu_{C_i})\}} \quad (3)$$

La función de fitness adaptada para minimizar el solapamiento es la siguiente:

$$\mathrm{F_{gs}(J)} = \begin{cases} \text{Sistema fuera de márgenes de seguridad:} \\ -(((\mathrm{C_I + C + E_c}) * (\mathrm{G_S})) + (\mathrm{T - t_d}) * (\mathrm{D + A})) \\ \\ \text{Sistema dentro de los márgenes de seguridad:} \\ -(\mathrm{C_I + C + E_c}) * (\mathrm{G_S}) \end{cases}$$

IV. Experimentos: problema del péndulo invertido, sin introducción de conocimiento

El péndulo invertido un problema clásico de control utilizado frecuentemente como benchmark en el campo del *Aprendizaje por Refuerzo* [9], [10]. Un bastón es colocado sobre un carro móvil, que se desplaza lateralmente sobre unos railes. Cualquier movimiento del carro tenderá a desestabilizar el bastón.

El objetivo es mover el carro a ambos lados manteniendo el péndulo en equilibrio, y sin colisionar con los límites de desplazamiento lateral del sistema. El controlador recibe la respuesta de refuerzo sólo después de que el péndulo se ha desequilibrado.

A. Variables del sistema

Para simplificar la exposición, en este problema hemos supuesto que todas las variables son accesibles en un único punto, coincidente con la situación del carrito en cada momento, lo que hace que la variable "Coste de implementación" sea 0.

B. Variables lógicas (Flags)

En este caso tendremos en cuenta sólo dos variables lógicas, o *Flags*, que indicarán la entrada del sistema en estados especiales. En ambos casos se trata de variables de tipo discreto, que sólo tomarán los valores 0, y 1. F_1 tomará valor 1 cuando el bastón se caiga, es decir cuando la variable e_1 esté fuera del rango [-1.57rad: 1.57rad]. F_2 tomará el valor 1 cuando el carrito abandone la zona de control [-2.4m:2.4m]. En caso de que cualquiera de los flags valga uno se activará una política de control especial consistente en detener la simulación y contabilizar el ejemplo como fallo.

TABLA I

PÉNDULO INVERTIDO. VARIABLES LÓGICAS.

Símbololo	Sigdo.	Tipo	Valor
F_1	Bastón desequilibrado	Discreto	0,1
F_2	Carrito fuera límites	Discreto	0,1

Las restricciones de funcionamiento, y las políticas de control especiales son las asociadas a estos dos flags.

C. Actuadores

La única acción de control en el caso del péndulo invertido consiste en aplicar una fuerza negativa o positiva sobre el carrito. El único actuador lo denominaremos A_1.

D. Objetivos

El objetivo final del controlador es llevar el sistema desde el estado inicial generado aleatoriamente a un estado de equilibrio descrito por una condición compleja. Si definimos una tolerancia para cada una de las variables relevantes, y las denominamos respectivamente TL_θ,$TL_{\dot\theta}$, TL_ρ, y $TL_{\dot\rho}$, el objetivo del controlador puede expresarse mediante la siguiente condición:

$$\begin{aligned} O_1 = \quad & (|\theta| < TL_\theta) \text{ AND } (|\dot\theta| < TL_{\dot\theta}) \\ & \text{AND } (|\rho| < TL_\rho) \text{ AND } (|\dot\rho| < TL_{\dot\rho}) \end{aligned} \tag{4}$$

E. Definición de la gramática.

La gramática que generará los individuos sometidos a evaluación se muestra en la figura 4

```
<Pendulo_Jerárquico> ::= Hacer (Pendulo No Controlado)
<Controlar_Péndulo>;
<Controlar_Péndulo> ::= SI <Zona_Central> ENTONCES
<Controlador_Central>
SINO <Controlador_Lateral>;
<Controlador_Central> ::= <Regla> <Lista_de_reglas>
|<Regla>;
<Controlador_Lateral> ::= <Regla> <Lista_de_Reglas>
|<Regla>;
<Regla> ::= SI <Antecedente> ENTONCES <Consecuente>;
<Antecedente> ::= <Lista_de_Condiciones>;
<Lista_de_Condiciones> ::= <Condición>
<Lista_de_Condiciones>
| <Condición>;
<Consecuente> ::= <Acción>;
<Zona_Central> := <Antecedente>;
<Acción> ::= FUERZA ES <Conjunto_Fuerza>;
<Conjunto_Fuerza> ::= NEGATIVA | CERO | POSITIVA ;
```

Fig. 4. Gramática del problema del péndulo invertido

F. Definición de la función de fitness.

Tal como la hemos definido en las secciones anteriores, la función de fitness gradual, descrita de forma genérica, se calcula de la siguiente manera:

$$\mathrm{F_g(J)} = \begin{cases} \text{Sistema fuera de márgenes de seguridad:} \\ -(\mathrm{C_I + C + E_c + (T - t_d) * (D + A)}) \\ \\ \text{Sistema dentro de márgenes de seguridad:} \\ -(\mathrm{C_I + C + E_c}) \end{cases}$$

Sea

- ρ_i la posición del carrito sobre el eje X en el instante i. En metros.
- θ_i el ángulo del bastón con respecto a 90º, en el instante i. En radianes
- a_i la acción de control generada en el instante i por el controlador. En Newtons.

Como se ha dicho, el coste de implementación (C_I) se considera nulo debido a que todas las variables son accesibles sin nedesidad de cableado alguno. La tabla II resume el significado de cada uno de los términos en el caso del péndulo invertido:

TABLA II

PÉNDULO INVERTIDO. FUNCIÓN DE FITNESS.

Cond. Seguridad	$\rho_i \in [-2{,}4, 2{,}4]$, $\theta_i \in [-1{,}57, 1{,}57]$
E_c	$\sum \lvert\rho_i\rvert + \lvert\theta_i\rvert$
C	$\sum \lvert a_i\rvert$
C_I	0
t_d	Desequilibrio
T	3000
D	2.4 m
A	10 N.

G. Modularidad.

La definición genérica del *Coeficiente de Modularidad*, en el caso del péndulo invertido, y siguiendo la gramática que hemos propuesto, se convierte en la siguiente expresión:

$$C_M(J) = \frac{|V_{C_c} \cup V_{C_l}| - |V_{C_c} \cap V_{C_l}|}{|V_{C_c} \cup V_{C_l}|} \tag{5}$$

donde

- C_c es el elemento descrito en la gramática como *Controlador_Central*, y
- C_l es el elemento descrito en la gramática como *Controlador_Lateral*

La función de fitness adaptada a esta nueva situación, tal como se especificó en secciones anteriores, es la siguiente:

$$\mathrm{F_{gm}(J)} = \begin{cases} \text{Sistema fuera de márgenes de seguridad:} \\ -(((\mathrm{C_I} + \mathrm{C} + \mathrm{E_c}) * (1 - \mathrm{C_M})) + (\mathrm{T} - \mathrm{t_d}) * (\mathrm{D} + \mathrm{A})) \\ \\ \text{Sistema dentro de los márgenes de seguridad:} \\ -(\mathrm{C_I} + \mathrm{C} + \mathrm{E_c}) * (1 - \mathrm{C_M}) \end{cases}$$

H. Solapamiento.

La definición general del *Grado de solapamiento* de un controlador jerárquico J, $G_S(J)$ se convierte, en el caso del péndulo invertido, y con las restricciones gramaticales que hemos indicado, en:

$$G_S(J) = \frac{\sum min\{(\mu_{C_{c_i}}, \mu_{C_{l_i}})\}}{\sum max\{(\mu_{C_{c_i}}, \mu_{C_{l_i}})\}} \tag{6}$$

En el caso que nos ocupa, el Péndulo Invertido, definiremos dos nuevos conceptos relacionados con la definición genérica de Solapamiento: El *Grado de Solapamiento Instantáneo* de un controlador jerárquico J en el instante i se definirá como:

$$G_{S_i}(J) = \frac{min\{(\mu_{C_{c_i}}, \mu_{C_{l_i}})\}}{max\{(\mu_{C_{c_i}}, \mu_{C_{l_i}})\}} \tag{7}$$

y nos dará una indicación de hasta qué punto la activación de uno de los controladores inhibe la activación de los demás, es decir, una forma de ver la independencia entre las particiones borrosas que delimita cada controlador de forma alternativa a la *Modularidad*.

Por *Solapamiento Medio por Periodo* entenderemos el valor medio del *Solapamiento Instantáneo* durante el periodo de simulación.

$$\overline{G_S(J)} = \frac{\frac{\sum min\{(\mu_{C_{c_i}}, \mu_{C_{l_i}})\}}{\sum max\{(\mu_{C_{c_i}}, \mu_{C_{l_i}})\}}}{T} \tag{8}$$

o, lo que es lo mismo:

$$\overline{G_S(J)} = \frac{\sum G_{S_i}(J)}{T} \tag{9}$$

donde T es la longitud del periodo de simulación, o el momento en el que el sistema está controlado, si fuera el caso.

La función de fitness modificada, incluyendo el *Solapamiento Medio por Periodo* es la siguiente:

$$\mathrm{F_{gs}(J)} = \begin{cases} \text{Sistema fuera de márgenes de seguridad:} \\ -(((\mathrm{C_I} + \mathrm{C} + \mathrm{E_c}) * (\overline{\mathrm{G_S}}(\mathrm{J}))) + (\mathrm{T} - \mathrm{t_d}) * (\mathrm{D} + \mathrm{A})) \\ \\ \text{Sistema dentro de los márgenes de seguridad:} \\ -(\mathrm{C_I} + \mathrm{C} + \mathrm{E_c}) * (\overline{\mathrm{G_S}}(\mathrm{J})) \end{cases}$$

I. Resultados.

En esta sección se recogen los parámetros más significativos del experimento realizado, y se comparan los resultados obtenidos mediante la *Función de Fitness Gradual*, con y sin *Solapamiento* y *Modularidad*.

Los parámetros del algoritmo, en todos los casos, se resumen en la tabla III.

TABLA III

PÉNDULO INVERTIDO. PARÁMETROS DEL ALGORITMO.

Tamaño Población	3000
Número Máximo Generaciones, G	50
Porcentaje de Mutación Aplicado	0.05
Función de Fitness	F_g, F_{gm}, F_{gs}
Criterio de terminación	G=50
Casos de prueba	100.
Casos de test	100.

La tabla V refleja la condición discriminante más frecuentemente inducida mediante cada una de las funciones de fitness con las que hemos trabajado.

Los aspectos más destacables sobre los resultados numéricos obtenidos se pueden concretar en que las funciones F_{gm} y F_{gs} aceleran la convergencia del proceso de aprendizaje, terminando el mismo en la condición central <Velocidad_Carrito_Nula> con mayor frecuencia que F_g, y reduciendo al mínimo la importancia de los casos no controlados pero tampoco desequilibrados. De estas dos, la función F_{gm} muestra un rendimiento superior en pruebas. La tabla IV recoge los resultados medios obtenidos en entrenamiento y test mediante las tres funciones de fitness evaluadas.

TABLA IV

PÉNDULO INVERTIDO. RESULTADOS.

Función de Fitness	Entr.	Test	nº reglas
F_g	80.04%	79.6%	5.72
F_{gm}	80.6%	84.84%	5.54
F_{gs}	82.52%	79.48%	5.46

TABLA V

CONDICIONES MÁS FRECUENTES.

F. de fitness	Condición	Frec.
F_g	<Velocidad_Carrito_Nula>	58%
F_{gm}	<Velocidad_Carrito_Nula>	80%
F_{gs}	<Velocidad_Carrito_Nula>	86%

V. PROBLEMA DEL PÉNDULO INVERTIDO: INTRODUCCIÓN DE CONOCIMIENTO EN EL PROCESO DE APRENDIZAJE.

A continuación resumimos el resultado de la repetición del experimento introduciendo distintos grados de conocimiento mediante modificaciones en la gramática. Analizaremos también las pérdidas de rendimiento que se produzcan en el algoritmo.

A. Selección de una función de adaptación.

A la vista de los resultados obtenidos en la sección anterior, la función de adaptación elegida para realizar el proceso de introducción de conocimiento en el algoritmo será la que hemos denominado *Fitness Gradual + modularidad*, o F_{gm}.

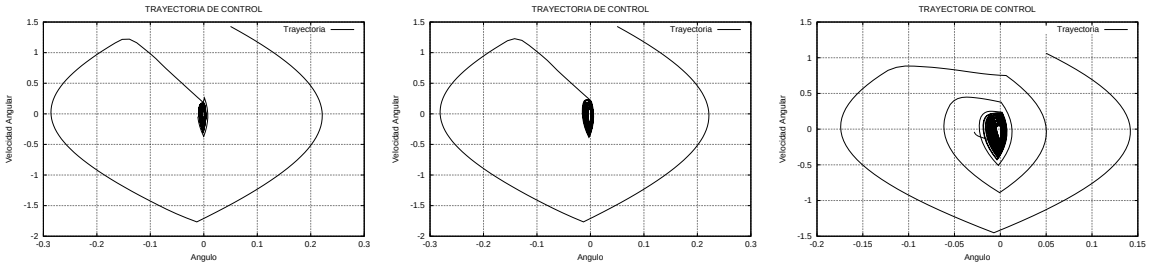

Fig. 5. Péndulo. Trayectorias de control.

B. *Utilización de una condición discriminante prefijada.*

Introduciremos en la gramática la definición estática de la condición <Zona_Central>. Pretendemos reflejar una filosofía de control consistente en aplicar un controlador distinto, probablemente menos intenso, cuando el sistema se encuentre en la zona próxima al equilibrio, y uno más enérgico para llevar el sistema a esa zona. La tabla VI refleja los resultados obtenidos mediante esta técnica.

TABLA VI
PÉNDULO INVERTIDO. RESULTADOS CON CONDICIÓN CENTRAL PREFIJADA.

Función de Fitness	Entr.	Test	nº reglas
F_{gm}	76.08%	71.28%	5.9

C. *Introducción de restricciones en la estructura de los controladores parciales.*

Manteniendo la condición discriminante del apartado anterior, limitaremos la forma de cada controlador parcial a tres únicas reglas de consecuentes prefijados, dejando al proceso de aprendizaje libertad para inducir únicamente las condiciones de aplicación de cada regla. La tabla VII refleja los resultados obtenidos mediante esta técnica.

TABLA VII
PÉNDULO INVERTIDO. RESULTADOS CON ESTRUCTURA DE BANCOS FIJADA.

Función de Fitness	Entr.	Test	nº reglas
F_{gm}	59%	53.11%	6

D. *Análisis de rendimiento.*

A la vista de los resultados, se puede afirmar que la introducción de una condición discriminante prefijada provoca un descenso en el rendimiento medio del algoritmo. Este descenso es mucho más acusado si se añaden restricciones a las estructuras de los bancos de reglas. En ambos casos, la aplicación de técnicas GA-P para mejorar la estructura profunda de los controladores inducidos resultó infructuosa, debido a la elección de unos parámetros adecuados en el inicio del proceso. La conclusión que se deriva de estos resultados es que cuando se incrementa la especificidad de las soluciones, el rendimiento medio del algoritmo de búsqueda se degrada. Esta pérdida de potencia puede, en algunos casos ser corregida mediante optimizaciones numéricas sobre los bancos de reglas, pero no siempre es posible. Será útil en los casos en los que la elección inicial de los conjuntos borrosos sea inadecuada.

VI. INDUCCIÓN DE LA POLÍTICA DE CONTROL DE UNA CENTRAL TÉRMICA.

A continuación se exponen algunos resultados preliminares obtenidos sobre el problema real que motivó el desarrollo de esta metodología.

A. *Descripción del problema.*

La central térmica de Aboño (Gijón), dispone de dos grupos térmicos que consumen una mezcla de carbón y gases siderúrgicos residuales procedentes de los hornos altos de la Factoría de Gijón de Aceralia. (Ver figura 6). En 1995 comenzó la remodelación de esta factoría, en la que se modernizaron sus dos hornos altos para aumentar su capacidad de producción de arrabio. Los consumos internos de gas en Aceralia no cambiaron significativamente, por lo que se incrementó considerablemente el caudal de los excedentes de gas. Este incremento supuso una modificación de las instalaciones de transporte de gas de horno alto en la planta de Aboño, con la consiguiente redefinición de la estrategia de control.

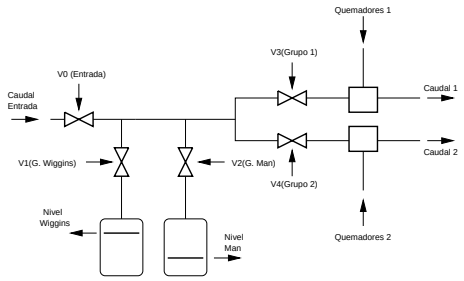

Fig. 6. Modelo de la Planta de Aboño.

De forma esquemática, la planta que se desea regular (ver figura 7) tiene las siguientes entradas:

- Caudal de entrada.
- Ángulo de la válvula de entrada al sistema.
- Ángulo de las válvulas de los dos gasómetros.
- Ángulo de las válvulas de entrada de los dos grupos.
- Número de quemadores en los grupos 1 y 2.

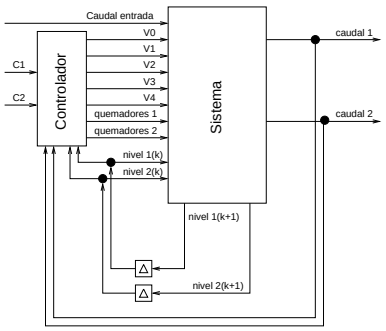

Fig. 7. Diagrama de bloques del controlador.

y dos salidas, que son:

- Caudal consumido en el grupo 1.
- Caudal consumido en el grupo 2.

B. *Variables del sistema:*

La tabla VIII recoge las variables relevantes para el controlador.

TABLA VIII
PLANTA DE ABOÑO. VARIABLES DEL SISTEMA

Variable	Significado
V0	Ángulo válvula principal
V1	Ángulo Wiggins
V2	Ángulo Mann
V3	Ángulo G1
V4	Ángulo G2
Q1	Quemadores G1
Q2	Quemadores G2
C1	Caudal G1
C2	Caudal G2
N1	Nivel Wiggins
N2	Nivel Mann
P	Política de reparto

B.1 Variables lógicas (Flags)

La tabla IX resume las variables lógicas identificadas en el sistema.

TABLA IX
PLANTA DE ABOÑO. VARIABLES LÓGICAS.

Sím.	Significado	Tipo	Val.
F_1	Wiggins descontrolado	Discreto	0,1
F_2	Mann descontrolado	Discreto	0,1
F_3	Presión Colector 1	Discreto	0,1
F_4	Presión Colector 2	Discreto	0,1
F_5	Velocidad Pistón 1	Discreto	0,1
F_6	Velocidad Pistón 2	Discreto	0,1

C. *Restricciones*

En el control de la alimentación de gas a una central térmica el control es multicriterio, y además hay algunas restricciones de seguridad, que deben incluirse en la función de fitness. Podemos sintetizar los objetivos del control en los siguientes puntos:

- El caudal de gas que se consume será, en promedio, el mismo qe el caudal que entra en el sistema. Se regula con las válvulas 3 y 4.
- El porcentaje del caudal consumido en cada rama es el seleccionado por el operador. Se regula con las válvulas 3 y 4.
- El gasómetro 1 ha de contener entre 20000 y 43000 m^3. el gasómetro 2 entre 12000 y 92000 m^3. Válvulas 1, 2, 3, y 4.
- La presión en ambos colectores del circuito dbe mantenerse entre 250 y 500 mm. ca., para que no se desceben los quemadores. Válvulas 3, 4 y política de control de 1y 2.
- Los pistones de los gasómetros no deben superar una determinada velocidad lineal, lo que quiere decir que el gasómetro 1 no puede proporcionar más de $90m^3/s$ ni el gasómetro 2 más de $210m^3/s$. Válvulas 1 y 2.
- Las válvulas han de moverse lo menos posible, por restricciones mecánicas del accionamiento.

D. *Políticas de control especiales*

Las políticas de control especiales provocadas por la activación de los flags son las siguientes:

- F_1: cierre de la válvula V1. Parada de la simulación.
- F_2 : cierre de la válvula V2. Parada de la simulación.
- F_3 : parada de la simulación. Quemadores del Grupo 1 descebados.
- F_4: parada de la simulación. Quemadores del Grupo 2 descebados.
- F_5: cierre de la válvula V1. Velocidad excesiva pistón 1.
- F_6: cierre de la válvula V2. Velocidad excesiva Pistón 2.

E. *Actuadores*

En este sistema de control existe un actuador por cada válvula, más uno por cada grupo, que indica el número de quemadores activos en ese momento.

F. *Definición de la función de fitness gradual.*

Tal como la hemos definido en las secciones anteriores, la función de fitness gradual, adaptada al caso de la Factoría de Aboño, se calcula de la siguiente manera:

$$\mathrm{F_g(J)} = \begin{cases} \text{Sistema fuera de márgenes de seguridad:} \\ -\infty \\ \\ \text{Sistema dentro de márgenes de seguridad:} \\ -(\mathrm{C_I} + \mathrm{C} + \mathrm{E_c}) \end{cases}$$

La tabla X resume el significado de cada uno de los términos en el caso de la planta:

G. *Resultados preliminares.*

En esta sección se recogen los resultados preliminares obtenidos al aplicar esta metodología a la Planta de Aboño. Los parámetros del algoritmo, en todos los casos, se resumen en la tabla XI. No se han incluido los resultados de las pruebas con modularidad

TABLA X
PLANTA DE ABOÑO. FUNCIÓN DE FITNESS.

C. de seguridad	
E_c	$\sum \lvert Erroresdeposicion \rvert$
C	$\sum \lvert \alpha_{V_i}(t+1) - \alpha_{V_i}(t) \rvert$
C_I	Coste de Cableado
T	longitud Periodo de Simulación

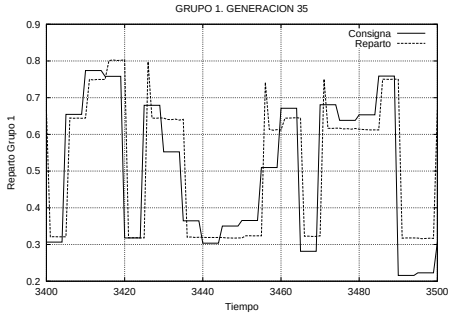

Fig. 8. Consigna y salida obtenida en la simulación de una central térmica. El control tiene calidad aceptable y su expresión puede comprenderse por un humano..

ni solapamiento, ni el coste de implementación en el fitness.

TABLA XI
PLANTA DE ABOÑO. PARÁMETROS DEL ALGORITMO.

Tamaño Población	3000
Número Máximo Generaciones	50
Porcentaje de Mutación Aplicado	0.05
Función de Fitness	F_g, F_{gm}, F_{gs}
Criterio de terminación	G=50.
Casos de prueba	100.
Casos de test	100.

TABLA XII
FACTORÍA DE ABOÑO. RESULTADOS.

Función de Fitness	Entrenamiento	Test
F_g	680.25	750.0

La figura 8 muestra el mejor controlador obtendo. Se puede observar que el sistema de aprendizaje consigue aproximar progresivamente la respuesta del sistema a la consigna o política de reparto entre grupos que se le indica, cumpliendo todas las restricciones de control y produciendo una expresión lingüística del sistema de control completo. Actualmente se están realizando nuevos experimentos, que incluyen la separación en módulos del sistema de control en función de la posición geográfica de cada elemento, y con diferentes niveles de conocimiento experto introducido en la gramática del sistema.

VII. CONCLUSIONES. TRABAJO FUTURO.

Existen numerosos trabajos en que se aplican algoritmos genéticos al aprendizaje supervisado de controladores borrosos basados en reglas. Sin embargo, hasta donde hemos podido conocer, los problemas del aprendizaje distal (a partir de un modelo) y, en general, del aprendizaje por refuerzo de estrategias de control han sido poco estudiados. En este trabajo se ha propuesto una forma de obtener expresiones lingüísticas (basadas en reglas borrosas) de estrategias de control a partir de un modelo del entorno y no a partir de ejemplos de funcionamiento. El algoritmo tuvo su origen en un problema real (actualmente no resuelto por completo, como se ha comentado) en el que consideraciones como el coste de cableado o la separación física de los elementos de control tienen importancia. Por esta razón, se ha buscado que el aprendizaje pueda ser adaptado de forma que premie las soluciones modulares con bajo coste de interconexión, y la autonomía de cada una de las partes del sistema. Adicionalmente, puede restringirse la búsqueda mediante la incorporación de conocimiento experto.

Es interesante observar que parece existir cierta correlación entre la frecuencia con que aparecen algunos bloques constructivos (como la condición central, en el caso del péndulo invertido) y algunas estrategias parciales proporcionadas por expertos. En un futuro próximo pretendemos profundizar en el estudio de este comportamiento, para decidir si es posible implementar un aprendizaje no supervisado con el que la propia gramática del sistema de control se autoorganice.

REFERENCIAS

[1] L.J. Lin, "Hierarchical learning of robot skills by reinforcement.," in *IEEE International Conference on Neural Networks*, 1993.

[2] R. Munos, "Finite-element methods with local triangulation refinement for continuous reinforcement learning problems.," in *European conference on Machine Learning*, 1997.

[3] S. Parr, R. Russel, "Reinforcement learning with hierarchies of machines," in *NIPS 97*, 1997.

[4] Precup D. Singh S Sutton, R.S, "Between mdps and semi-mdps: learning, planning, and representing knowledge at multiple temporal scales," 1998, Technical Report 98-74. University of Massachusets, Amherts, MA.

[5] Andreas. Geyer-Schulz, *Fuzzy Rule-Based Expert Systems and Genetic Machine Learning.*, Physica-Verlag., second edition, 1997.

[6] L. Sánchez I. Couso and J.A. Corrales, "Combining g.p. operators with s.a. search to evolve fuzzy rule based classifiers," *Information Sciences, 1-4*, vol. 136, pp. 175–191, 2001.

[7] Santiago Garcia, Fermin Gonzalez, and Luciano Sanchez, "Evolving fuzzy rule based classifiers with GAP: A grammatical approach," in *Genetic Programming, Proceedings of EuroGP'99*, Riccardo Poli, Peter Nordin, William B. Langdon, and Terence C. Fogarty, Eds., Goteborg, Sweden, 26-27 May 1999, vol. 1598 of *LNCS*, pp. 203–210, Springer-Verlag.

[8] D. J. Howard L. M., D'Angelo, "The ga-p: A genetic algorithm and genetic programming hybrid," *IEEE Expert.*, vol. 10, pp. 11–15, 1995.

[9] Mikkulainen R. Moriarty, D.E., "Efficient reinforcement learning trough simbiotic evolution," *Machine Learning.*, vol. 22, pp. 11–32, 1996.

[10] Mikkulainen R. Moriarty, D.E., "Forming neural networks trough efficient and adaptive co-evolution," *Evolutionary Computation*, vol. 5, pp. 373–399, 1998.